Serial No.: 09/715,778

PATENT APPLICATION
Docket No.: NC 84,779

AMENDMENTS TO THE CLAIMS

This listing of claims will replace all prior versions and listings of claims in the application.

1. (currently amended) An apparatus comprising:
a processor capable of simultaneous execution of two or more threads of instructions, where said processor comprises:
at least one resource unit capable of being assigned to two or more of the threads;
a priority register to store thread information for the [[P]] threads, the thread information including a priority code ~~P priority codes~~ corresponding to each thread ~~the P threads~~, at least one of the [[P]] threads requesting use of the ~~at least one~~ resource unit; and
a priority selector coupled to the priority register to generate an assignment signal to assign the at least one resource unit to the requesting thread ~~at least one of the P threads~~ according to the [[P]] priority codes.

2. (currently amended) The apparatus of claim 1 wherein the at least one resource unit is one of an instruction fetch unit, ~~an instruction buffer,~~ a memory locking unit, a load unit, a store unit, an input/output unit, a peripheral unit interface, and a functional unit.

3. (original) The apparatus of claim 2 wherein the functional unit is one of an arithmetic unit, a logic unit, and an arithmetic and logic unit.

4. (original) The apparatus of claim 1 further comprising:
an instruction multiplexer coupled to the priority selector to pass instructions stored in a plurality of instruction registers to execution units according to the assignment signal.

5. (currently amended) The apparatus of claim 1 further comprising:
a priority assignor coupled to the priority register to set the thread information including at least one of the [[P]] priority codes corresponding to the at least one of the [[P]] threads in response to a start instruction from an instruction decoder and dispatcher.

6. (currently amended) The apparatus of claim 5 wherein the priority assignor sets an active flag in the priority register corresponding to the at least one of the [[P]] threads in response to the start instruction.

7. (currently amended) The apparatus of claim 6 wherein resets the active flag in the priority register corresponding to the at least one of the [[P]] threads in response to a quit instruction from the instruction decoder and dispatcher.

8. (currently amended) The apparatus of claim 1 wherein the priority selector assigns the at least one resource unit to the at least one of the [[P]] threads if the at least one of the [[P]] threads is not served and the at least one resource unit is free.

9. (currently amended) The apparatus of claim 8 wherein the at least one of the [[P]] threads has highest priority code among a set of ready threads ~~of the P threads~~.

10. (currently amended) The apparatus of claim 8 wherein the priority selector iteratively assigns resource units to threads in the set of ready threads ~~of the P threads~~ according to the corresponding priority codes and resource availability until the set becomes empty.

11. (original) A method comprising:
executing two or more threads of instructions simultaneously in a processor;
storing thread information for the threads ~~P threads~~, the thread information including a priority code ~~P priority codes~~ corresponding to each thread ~~the P threads~~, at least one of the P threads requesting use of at least one resource unit capable of being assigned to two or more of the threads; and
generating an assignment signal to assign the at least one resource unit to the requesting thread ~~at least one of the P threads~~ according to the [[P]] priority codes.

12. (currently amended) The method of claim 11 wherein the at least one resource unit is one of an instruction fetch unit, ~~an instruction buffer,~~ a memory locking unit, a load unit, a store unit, an input/output unit, a peripheral unit interface, and a functional unit.

13. (original) The method of claim 12 wherein the functional unit is one of an arithmetic unit, a logic unit, and an arithmetic and logic unit.

14. (original) The method of claim 11 further comprising:
passing instructions stored in a plurality of instruction registers to execution units according to the assignment signal.

15. (currently amended) The method of claim 11 further comprising:
setting the thread information including at least one of the [[P]] priority codes corresponding to the at least one of the [[P]] threads in response to a start instruction from an instruction decoder and dispatcher.

16. (currently amended) The method of claim 15 wherein setting the thread information comprises setting an active flag in the priority register corresponding to the at least one of the [[P]] threads in response to the start instruction.

17. (currently amended) The method of claim 16 wherein setting the thread information comprises resetting the active flag in the priority register corresponding to the at least one of the [[P]] threads in response to a quit instruction from the instruction decoder and dispatcher.

18. (currently amended) The method of claim 11 wherein generating the assignment signal comprises generating the assignment signal to assign the at least one resource unit to the at least one of the [[P]] threads if the at least one of the [[P]] threads is not served and the at least one resource unit is free.

19. (currently amended) The method of claim 18 wherein the at least one of the [[P]] threads has highest priority code among a set of ready threads ~~of the P threads~~.

20. (original) The method of claim 1 wherein generating the assignment signal comprises iteratively assigning resource units to threads in the set of ready threads ~~of the P threads~~ according to the corresponding priority codes and resource availability until the set becomes empty.

21. (currently amended) A processor capable of simultaneous execution of two or more threads of instructions comprising:

at least one a resource unit to provide resource for use by the [[P]] threads, capable of being assigned to two or more of the threads; and

a resource prioritizer coupled to the resource unit to prioritize resource utilization, the resource prioritizer comprising:

a priority register to store thread information for the [[P]] threads, the thread information including a priority code ~~P priority codes~~ corresponding to each thread ~~the P threads~~, at least one of the [[P]] threads requesting use of the ~~at least one~~ resource unit, and

a priority selector coupled to the priority register to generate an assignment signal to assign the at least one resource unit to the requesting thread ~~at least one of the P threads~~ according to the [[P]] priority codes.

22. (currently amended) The processor of claim 21 wherein the at least one resource unit is one of an instruction fetch unit, ~~an instruction buffer,~~ a memory locking unit, a load unit, a store unit, an input/output unit, a peripheral unit interface, and a functional unit.

23. (original) The processor of claim 22 wherein the functional unit is one of an arithmetic unit, a logic unit, and an arithmetic and logic unit.

24. (original) The processor of claim 21 the resource prioritizer further comprising:
an instruction multiplexer coupled to the priority selector to pass instructions stored in a plurality of instruction registers to execution units according to the assignment signal.

25. (currently amended) The processor of claim 21 wherein the resource prioritizer further comprising:
a priority assignor coupled to the priority register to set the thread information including at least one of the [[P]] priority codes corresponding to the at least one of the [[P]] threads in response to a start instruction from an instruction decoder and dispatcher.

26. (currently amended) The processor of claim 25 wherein the priority assignor sets an active flag in the priority register corresponding to the at least one of the [[P]] threads in response to the start instruction.

27. (currently amended) The processor of claim 26 wherein resets the active flag in the priority register corresponding to the at least one of the [[P]] threads in response to a quit instruction from the instruction decoder and dispatcher.

28. (currently amended) The processor of claim 21 wherein the priority selector assigns the at least one resource unit to the at least one of the [[P]] threads if the at least one of the P threads is not served and the at least one resource unit is free.

29. (currently amended) The processor of claim 28 wherein the at least one of the [[P]] threads has highest priority code among a set of ready threads ~~of the P threads~~.

30. (currently amended) The processor of claim 28 wherein the priority selector iteratively assigns resource units to threads in the set of ready threads ~~of the P threads~~ according to the corresponding priority codes and resource availability until the set becomes empty.